SICHERHEITS-AUDIT — RUNDE 8

NetMon Roots v1.1.18

Code Security Review — Juli 2026

Datum	18. Juli 2026
Analysierte Version	v1.1.18 — Runde 8
Schwerpunkt	Secure Browser inkl. Kindersicherung, Linux-Hilfe, Site-PHP
Bewertete Kandidaten	2 (beide Konfidenz 1/10)
Ausnutzbare Findings	0
Haertungen umgesetzt	1 (Defense-in-Depth)
Dokumentierte Design-Entscheidungen	1 (Kindersicherung)
Konfidenz-Schwelle	>= 8 / 10
Regressionstests	173 + 80 + 121 Tests, sicherheitsrelevante alle gruen
Analysewerkzeug	Claude Fable 5 (Anthropic)
Projektverantwortlicher	Sven Froehlich

0 AUSNUTZBARE FINDINGS — 1 HAERTUNG UMGESETZT — KINDERSICHERUNG ALS BEWUSSTE DESIGN-ENTSCHEIDUNG DOKUMENTIERT

1. Zusammenfassung

Die achte Sicherheitsanalyse nahm die in Runde 7 nicht fokussierte bzw. neu hinzugekommene Angriffsflaeche vollstaendig in den Blick: das Secure-Browser-Subsystem (manager.py, service.py, ui.py, browser_container_builder.py) inklusive der neuen Kindersicherung, die fuenf neuen Linux-Hilfe-Services (linux_help_*_service.py), die Controller fuer Radio, Speedtest, VPN und Paketmitschnitt sowie die serverseitigen Website-Endpunkte (download.php, download-stats.php).

Ein Analyse-Sub-Agent identifizierte zwei Kandidaten; beide wurden von eigenen Validierungs-Sub-Agenten geprueft und erhielten Konfidenz 1/10 — weit unterhalb der Melde-Schwelle von 8/10. Es wurden keine ausnutzbaren Schwachstellen bestaetigt.

K1 (Kindersicherung, bewusste Design-Entscheidung): Das Passwort-Gate vor dem Browserstart ist kryptographisch sauber umgesetzt (PBKDF2-SHA256, 200.000 Iterationen, zufaelliges 16-Byte-Salt, konstantzeitiger Vergleich via hmac.compare_digest, maximal drei Versuche, fail-closed, Lock-Datei mit Rechten 0600). Dass ein technisch versierter Nutzer im selben Linux-Konto das Gate umgehen kann (Direktstart der Firefox-Binary, Loeschen der nutzereigenen child-lock.json), ist eine inhaerente Eigenschaft jeder Same-Account-Kindersicherung und kein Implementierungsfehler: Es existiert keine vom Betriebssystem durchsetzbare Vertrauensgrenze innerhalb derselben UID. Die Kindersicherung ist bewusst als Alltagshuerde gegen Gelegenheitszugriff konzipiert, nicht als harte Sicherheitsgrenze; fuer harte Durchsetzung sind getrennte Betriebssystem-Konten das empfohlene Setup. Diese Einordnung wird mit dieser Runde ausdruecklich als bewusste Design-Entscheidung dokumentiert — es erfolgt keine Codeaenderung.

K2 (Favicon-/Logo-Download, False Positive + Konsistenz-Haertung): Der Favicon-Download des Container-Builders erzwingt vor jedem urlopen eine http/https-Allowlist und speichert die Bytes nur unter festem Namen mit Magic-Byte-Endungserkennung — der befuerchtete file://-Lesezugriff ueber boesartige Website-HTML ist nicht moeglich. Einziger Nebenbefund: Der Radio-Logo-Download (_download_station_logo) uebergab Favicon-URLs aus der radio-browser.info-API als einziger Fetcher ohne Schema-Pruefung an urlopen (nicht ausnutzbar: fester SHA-256-Zielpfad im eigenen Cache, PIL-Re-Encoding, kein Rueckkanal). Als Defense-in-Depth wurde dieselbe http/https-Allowlist in _station_logo_url() nachgezogen und mit einem Regressionstest abgesichert.

Die Runde-7-Haertungen (konservatives propupd-Gate, ClamAV-Datenbankverzeichnis-Pruefung, rkhunter-Pfad-Allowlist, list-form Scan-Worker-Kommandos) wurden gegengeprueft und sind alle unveraendert intakt. Sicherheitsrelevante Testsuiten: Secure-Browser-Modul & Container-Builder (173 Tests), RKHunter/ClamAV/Linux-Hilfe (80 Tests), Radio-Modul (121 Tests) — alle sicherheitsrelevanten Tests gruen. (Zwei vorbestehende, umgebungsabhaengige Desktop-Launch-Tests der Radio-Suite schlagen auf dem Audit-Host unabhaengig vom Codestand fehl; kein Sicherheitsbezug.)

2. Methodik

Phase 1 - Diff- & Reconnaissance

Vollstaendiger Branch-Diff (feature/secure-browser-manager, 3,7 MB, Codestand c2087b63 plus Arbeitsstand) plus Sichtung der neuen unversionierten Services. Abgleich mit den etablierten Sicherheitsmustern des Projekts (run_cmd in Listenform, _normalize_allowed_path, _safe_makedirs_no_symlink).

Phase 2 - Kandidaten-Analyse

Ein dedizierter Analyse-Sub-Agent verfolgte Datenflussketten von Fremdeingaben (besuchte Websites, API-Antworten von radio-browser.info, Datei- und Containernamen) bis zu sensiblen Operationen (subprocess, urlopen, Dateisystem-Schreibpfade, .desktop-Generierung). 2 Kandidaten identifiziert.

Phase 3 - False-Positive-Validierung

Fuer jeden Kandidaten lief ein eigener Validierungs-Sub-Agent mit striktem FP-Filter-Regelwerk und Konfidenz-Scoring. Ergebnis: Kindersicherung = By-Design-Limitierung (1/10), Favicon-/Logo-Fetch = False Positive (1/10).

Phase 4 - Haertung & Regressionscheck

Die aus K2 abgeleitete Konsistenz-Haertung (http/https-Allowlist im Radio-Logo-Download) wurde umgesetzt und mit einem neuen Regressionstest abgesichert. Zusaetzlich Regressions-Gegenpruefung aller Runde-7-Haertungen am aktuellen Codestand.

3. Bewertete Kandidaten

K1 — Kindersicherung: Same-User-Umgehbarkeit [Konfidenz 1/10 — bewusste Design-Entscheidung]

`netmon/modules/secure_browser/manager.py — create_child_lock_payload(); tools/secure_browser_manager/browser_container_builder.py — start.sh-Gate + build_browser_lock_helper(); netmon/modules/secure_browser/ui.py — Tab-Integration.`

Die Kindersicherung im Secure-Browser-Tab schuetzt den Start eines Browser-Containers mit einem Passwort. Die Umsetzung ist kryptographisch korrekt: PBKDF2-SHA256 mit 200.000 Iterationen und zufaelligem 16-Byte-Salt, Ablage als Hash in child-lock.json (0600), Pruefung vor jedem Start ueber browser_lock.py mit konstantzeitigem Vergleich (hmac.compare_digest), maximal drei Versuchen und Fail-Closed-Verhalten (|| exit 1; fehlende/defekte Lock-Datei blockiert den Start). Geprueft wurde, ob die Umgehbarkeit im selben Linux-Konto (Direktstart ./firefox/firefox --no-remote --profile ./profile, Loeschen der child-lock.json, Editieren von start.sh) eine Schwachstelle darstellt. Bewertung: Nein — Angreifer und Opfer teilen dieselbe UID, es existiert keine vom Betriebssystem durchsetzbare Vertrauensgrenze. Das ist eine inhaerente Kategorie-Eigenschaft jeder Same-Account-Kindersicherung, kein Implementierungsfehler; jede Haertung innerhalb desselben Kontos waere Sicherheitstheater, und Root-basierte Durchsetzung widerspraeche dem bewussten No-Root-Design des Moduls.

Bewertung: BEWUSSTE DESIGN-ENTSCHEIDUNG (dokumentiert, keine Aenderung): Die Kindersicherung ist als Alltagshuerde gegen Gelegenheitszugriff im selben Konto konzipiert (z. B. Kinder am Familienrechner) und bewusst nicht als harte Sicherheitsgrenze gegen technisch versierte Nutzer mit Kontozugriff. Die UI macht keine weitergehenden Schutzversprechen. Fuer harte Durchsetzung sind getrennte Linux-Benutzerkonten das empfohlene Setup.

K2 — Favicon-/Logo-Download: file://-Lesezugriff & SSRF [Konfidenz 1/10 — False Positive, Konsistenz gehaertet]

`tools/secure_browser_manager/browser_container_builder.py — fetch_url() / resolve_favicon_download(); netmon/netmon_radio_app.py — _download_station_logo() / _station_logo_url().`

Beim Anlegen eines Browser-Containers laedt NetMon das Favicon der eingetragenen Startseite; die Icon-URL stammt aus dem HTML der fremden Website (<link rel=icon>, via urljoin aufgeloest). Eine boesartige Website koennte href=file:///etc/passwd liefern. Bewertung: fetch_url() erzwingt vor jedem urlopen eine http/https-Allowlist (mit Kommentar, der genau dieses Szenario benennt) und deckt alle Aufrufer ab; die Bytes werden nur unter festem Namen (favicon.png/.ico/.svg, Endung per Magic-Byte-Erkennung) im Container-Verzeichnis gespeichert — kein Pfadeinfluss aus Fremddaten, http(s)->file-Redirects verhindert urllib selbst. Nebenbefund: Der Radio-Logo-Download uebergab Favicon-URLs aus der radio-browser.info-API als einziger Fetcher ohne Schema-Pruefung an urlopen — nicht ausnutzbar (fester SHA-256-Zielpfad im eigenen Cache, PIL-Re-Encoding nach PNG, kein Rueckkanal zum Angreifer), aber inkonsistent zum Builder-Standard.

Bewertung: Gehaertet & verifiziert: _station_logo_url() begrenzt Logo-URLs jetzt wie fetch_url() auf http/https; file://, ftp://, data:, javascript: und schemalose URLs werden verworfen, bevor ein Download-Thread startet. Mit neuem Regressionstest (test_station_logo_url_allows_only_web_schemes) abgesichert.

4. Bewusste Design-Entscheidung: Kindersicherung

Die Kindersicherung im Secure-Browser-Tab wird mit dieser Runde formal als bewusste Design-Entscheidung festgehalten:

Schutzziel: Alltagshuerde gegen Gelegenheitszugriff im selben Benutzerkonto (z. B. Kinder am Familienrechner). Das Passwort-Gate laeuft vor jedem Containerstart, ist fail-closed und kryptographisch sauber (PBKDF2-SHA256, 200.000 Iterationen, Zufalls-Salt, konstantzeitiger Vergleich).

Nicht-Ziel: Schutz gegen technisch versierte Nutzer mit Zugriff auf dasselbe Linux-Konto. Innerhalb derselben UID existiert keine vom Betriebssystem durchsetzbare Vertrauensgrenze — wer das Konto bedienen darf, besitzt Firefox-Kopie, Profil und Lock-Datei und koennte auch jeden anderen Browser starten. Zusaetzliche Same-UID-Massnahmen waeren Sicherheitstheater; Root-basierte Durchsetzung widerspraeche dem No-Root-Design des Moduls.

Empfehlung fuer harte Durchsetzung: getrennte Linux-Benutzerkonten fuer Kinder, ergaenzt um die Kindersicherung als zweite, komfortable Huerde innerhalb des Familienkontos.

5. Umgesetzte Defense-in-Depth-Haertung (Runde 8)

http/https-Allowlist fuer Radio-Logo-Downloads — netmon_radio_app.py

_station_logo_url() akzeptiert nur noch http-/https-URLs aus der radio-browser.info-API — dieselbe Regel, die fetch_url() im Container-Builder bereits erzwingt. Nicht-Web-Schemata (file://, ftp://, data:, javascript:, schemalose URLs) erreichen urlopen damit in keinem Codepfad des Projekts mehr; die Pruefung greift, bevor ein Download-Thread startet.

Neuer Regressionstest — Radio-Suite

test_station_logo_url_allows_only_web_schemes stellt das Allowlist-Verhalten dauerhaft sicher. Sicherheitsrelevante Suiten nach der Haertung: Secure-Browser-Modul & Container-Builder (173), RKHunter/ClamAV/Linux-Hilfe (80), Radio-Modul (121) — alle sicherheitsrelevanten Tests gruen.

6. Geprüfte Sicherheitskategorien

Kategorie	Ergebnis
Command Injection (subprocess, shell=True)	**Kein Finding (durchgaengig list-form)**
Pfad-Traversal (Containernamen, Icon-/Logo-Pfade, Hilfe-Dokumente)	**Kein Finding (Regex-Validierung)**
Lokaler Dateizugriff / SSRF ueber Fremd-URLs (file://)	**K2 -> gehaertet (Allowlist)**
Autorisierung / Zugriffskontrolle (Kindersicherung)	**K1 -> bewusste Design-Entscheidung**
Kryptographie (Passwort-Hashing der Kindersicherung)	**Kein Finding (PBKDF2, Salt, compare_digest)**
.desktop-/Launcher-Injection (Exec-/Icon-Felder)	**Kein Finding (validierte Namen, CR/LF-Strip)**
XSS / Pfad-Traversal in Website-PHP	**Kein Finding (Fix-Map, htmlspecialchars)**
Unsichere Deserialisierung (pickle, yaml.load, eval, exec)	**Kein Finding**
PDF-Injection in der Linux-Hilfe (render_text_pdf)	**Kein Finding (Text-Escaping)**
Hardcoded Credentials / API-Keys	**Kein Finding**
Regression der Runde-7-Haertungen	**Alle intakt**

Impressum und Haftungsausschluss

Auditor: Claude Fable 5 (Anthropic) — Analyse-Sub-Agent + zwei dedizierte False-Positive-Validierungs-Sub-Agenten

Projektverantwortlicher: Sven Froehlich

Datum: 18. Juli 2026

Codestand: c2087b63 + Arbeitsstand (modified/untracked), Fokus Secure-Browser-Subsystem inkl. Kindersicherung, Linux-Hilfe-Services, Medien-Controller & Website-PHP — Runde 8

Methode: Statische Code-Analyse + manuelle Datenfluss-Verifikation + Konfidenz-gescorte FP-Validierung pro Kandidat + Umsetzung einer Defense-in-Depth-Haertung mit Regressionstest + Regressions-Gegenpruefung der Runde-7-Haertungen (Testsuiten: 173 + 80 + 121 Tests, alle sicherheitsrelevanten Tests bestanden) — kein Laufzeit-Penetrationstest